IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

APPLICATION FOR UNITED STATES PATENT

Title:	PHASE LOCK LOOPED OSCILLATORY NEUROCOMPUTER
Inventor(s):	FRANK C. HOPPENSTEADT PARADISE VALLEY, ARIZONA EUGENE M. IZHIKEVICH SAN DIEGO, CALIFORNIA
Attorneys:	Thomas D. MacBlain GALLAGHER & KENNEDY, P.A. 2575 East Camelback Road Phoenix, Arizona 85016-9225 Telephone: (602) 530-8000

PHASE-LOCKED LOOP OSCILLATORY NEUROCOMPUTER

The present patent application claims priority rights on U.S. Provisional Patent Application Serial No. 60/178,640 filed January 28, 2000.

FIELD OF THE INVENTION

The present invention relates, in general, to computational devices and, more particularly, to a recurrent neural network computer based on phase modulation of Phase-Locked Loop (PLL) nonlinear oscillators.

BACKGROUND OF THE INVENTION

Neural network computers, are biologically inspired, that is, they are composed of elements that perform in a manner analogous to the most elementary functions of the biological neuron. In one methodology, a neural network computer is composed of a number (n) of processing elements that may be switches or nonlinear amplifiers. These elements are then organized in a way that may be related to the anatomy of the brain. The configuration of connections, and thus communication routes, between these elements represents the manner in which the neural network computer will function, analogous to that of a program performed by digital computers. Despite this superficial resemblance, such artificial neural networks exhibit a surprising number of the brain's characteristics. For example, they learn from experience, generalize from previous examples to new ones, and abstract essential characteristics from inputs containing irrelevant data. Unlike a von Neumann computer, such a neural network computer does not execute a list of commands (a program). Rather, it performs pattern recognition and associative recall via self-organization of connections between elements.

Artificial neural networks can modify their behavior in response to their environment. Shown a set of inputs (perhaps with desired outputs), they self-adjust to produce consistent responses. A network is trained so that application of a set of inputs produces the desired (or at least consistent) set of outputs. Each such input (or output) set is referred to as a vector. Training can be accomplished by sequentially applying input vectors, while adjusting network weights according to a predetermined procedure, or by setting weights a priori. During training,

the network weights gradually converge to values such that each input vector produces the desired output vector.

Because of their ability to simulate the apparently oscillatory nature of brain neurons, oscillatory neural network computers are among the more promising types of neural network computers. Simply stated, the elements of an oscillatory neural network computer consist of oscillators rather than amplifiers or switches. Oscillators are mechanical, chemical or electronic devices that are described by an oscillatory signal (periodic, quasi-periodic, almost periodic function, etc.) Usually the output is a scalar function of the form $V(\omega t+\phi)$ where V is a fixed wave form (sinusoid, saw-tooth, or square wave), ω is the frequency of oscillation, and ϕ is the phase deviation (lag or lead).

Recurrent neural networks have feedback paths from their outputs back to their inputs. The response of such networks is dynamic in that after applying a new input, the output is calculated and fed back to modify the input. The output is then recalculated, and the process is repeated again and again. Ideally, successive iterations produce smaller and smaller output changes until eventually the outputs become steady oscillations or reach a steady state. Although these techniques have provided a means for recognizing signals, to date they have not been able to do so using associative memory.

Accordingly, a need exists for a neural network computer with fully recurrent capabilities and a method that incorporates the periodic nature of neurons in the pattern recognition process.

SUMMARY OF THE INVENTION

In accordance with the present invention, an oscillatory neural network computer is disclosed that exhibits pattern recognition using the phase relationships between a learned pattern and an incoming pattern, i.e., the pattern to be recognized.

In one aspect of the present invention, the oscillatory neural network computer comprises a weighting circuit having phase-based connection strengths. A plurality of phase-locked loop circuits are operably coupled to the weighting circuit.

In another aspect of the present invention, a method for programming an oscillatory neural network computer is provided wherein programming comprises encoding connection coefficients of the oscillatory neural network computer in accordance with phase relationships of a pattern to be learned.

In yet another aspect of the present invention, a method for recognizing an incoming pattern using an oscillatory neural network computer is provided wherein the method comprises using the phase deviation between a learned pattern and the incoming pattern to create an output signal indicative of the learned pattern.

BRIEF DESCRIPTION OF THE DRAWING

FIG. 1 schematically illustrates an oscillatory neural network computer in accordance with an embodiment of the present invention;

FIG. 2 schematically illustrates a weighting element in accordance with an embodiment of the present invention;

FIG. 3 illustrates patterns to be memorized by the oscillatory neural network computer;

FIG. 4 illustrates a pattern to be recognized in accordance with an embodiment of the present invention;

FIG. 5 is a plot of the frequency and the phase portions of output signals of the oscillatory neural network computer in accordance with the present invention;

FIG. 6 is a plot of the frequency and the phase portions of other output signals of the neural network computer in accordance with the present invention;

FIG. 7 illustrates the output signals to the pixels during the recognition process in accordance with the present invention; and

FIG. 8 illustrates the products of the output signals to the pixels of FIG. 7.

DETAILED DESCRIPTION

The oscillatory neural network computer of the present invention learns or memorizes information in terms of periodic waveforms having an amplitude, a frequency, and a phase. This information is encoded as connection strengths, $S_{k,j}$, using a learning rule such as, for

example, the Hebbian learning rule. The connection strengths, in combination with phase-locked loop circuitry, are used to recognize information from signals transmitted to the oscillatory neural network computer.

Copending U.S. patent application Serial No. PCT/US99/26698, entitled "OSCILLATORY NEUROCOMPUTER WITH DYNAMIC CONNECTIVITY" and filed November 12, 1999 by Frank Hoppensteadt and Eugene Izhikevich is hereby incorporated herein by reference in its entirety.

FIG. 1 schematically illustrates an oscillatory neural network computer 20 in accordance with an embodiment of the present invention. Oscillatory neural network computer 20 comprises a weighting network 21 coupled in a feedback configuration to a plurality of phase-locked loop circuits 25_1, ..., 25_{N-1}, 25_N. Oscillatory neural network computer 20 has output terminals OUT_1, OUT_2, ..., OUT_{N-1}, OUT_N for transmitting output signals $V(\theta_1)$, $V(\theta_2)$, ..., $V(\theta_{N-1})$, $V(\theta_N)$, respectively, where the output signals $V(\theta_1)$, $V(\theta_2)$, ..., $V(\theta_{N-1})$, $V(\theta_N)$ have equal frequencies and constant, but not necessarily zero, phase relationships. More particularly, weighting network 21 includes a plurality $C_{1,1}$, $C_{1,2}$, ..., $C_{N,N}$ of weighting circuits, a plurality 31_1, ..., 31_N of adder circuits, and a plurality 35_1, ..., 35_N of bandpass filter circuits. Weighting circuits $C_{1,1}$, $C_{1,2}$, ..., $C_{N,N}$ are configured as an N x N matrix. By way of example, weighting network 21 is a symmetric matrix of weighting circuits where each weighting element is coupled for transmitting an output signal having a connection strength, $S_{k,j}$, associated therewith. The connection strength is also referred to as the connection weight, the connection coefficient, the interunit connection strength, or simply the strength or weight. It should be understood that reference number 25 is used to collectively identify the plurality of phase-locked loop circuits, reference number 31 is used to collectively identify the plurality of adder circuits, and reference number 35 is used to collectively identify the plurality of bandpass filter circuits. The subscript notation (1, 2, ..., N) has been appended to reference numbers 25, 31, and 35 to identify individual phase-locked loop circuits, adder circuits, and bandpass filter circuits, respectively. Further, the weighting circuits may also be referred to as connectors or weighting elements.

Weighting circuits $C_{1,1}$, $C_{1,2}$, ..., $C_{1,N-1}$, $C_{1,N}$ are connected to the input terminals of adder circuit 31_1 and to the respective output terminals OUT_1, OUT_2, ..., OUT_{N-1}, OUT_N.

Weighting circuits $C_{2,1}$, $C_{2,2}$, ..., $C_{2,N-1}$, $C_{2,N}$ are connected to the input terminals of adder circuit 31_2, to the input terminals of the respective weighting elements $C_{1,1}$, $C_{1,2}$, ..., $C_{1,N-1}$, $C_{1,N}$ and to the respective output terminals OUT_1, OUT_2, ..., OUT_{N-1}, OUT_N. Weighting circuits $C_{N-1,1}$, $C_{N-1,2}$, ..., $C_{N-1,N-1}$, $C_{N-1,N}$ are connected to the input terminals of adder circuit 31_{N-1} and to the respective output terminals OUT_1, OUT_2, ..., OUT_{N-1}, OUT_N. Weighting circuits $C_{N,1}$, $C_{N,2}$, ..., $C_{N,N-1}$, $C_{N,N}$ are connected to the input terminals of adder circuit 31_N and to the respective output terminals OUT_1, OUT_2, ..., OUT_{N-1}, OUT_N.

Further, initialization input terminals IN_1, IN_2, ..., IN_{N-1}, IN_N are coupled to initialization input terminals of adder circuits 31_1, 31_2, ..., 31_{N-1}, 31_N, respectively. The output terminals of adder circuits 31_1, 31_2, ..., 31_{N-1}, 31_N are connected to the input terminals of bandpass filter circuits 35_1, 35_2, ..., 35_{N-1}, 35_N, respectively.

The output terminals of bandpass filter circuits 35_1, 35_2, ..., 35_{N-1}, 35_N are connected to the input terminals of phase-locked loop circuits 25_1, 25_2, ..., 25_{N-1}, 25_N, respectively. The output terminals of phase-locked loop circuits 25_1, 25_2, ..., 25_{N-1}, 25_N are connected to the respective output terminals OUT_1, OUT_2, ..., OUT_{N-1}, OUT_N.

FIG. 2 schematically illustrates an embodiment of a weighting circuit ($C_{1,1}$, $C_{1,2}$, ..., $C_{N,N}$) in accordance with the present invention. In this embodiment, weighting circuits $C_{1,1}$, $C_{1,2}$, ..., $C_{N,N}$ comprise a linear amplifier 23 having an input terminal connected to a respective output terminal OUT_1, OUT_2, ..., OUT_{N-1}, OUT_N. An output terminal of linear amplifier 23 is connected to an input terminal of phase shift circuit 24. An output terminal of phase shift circuit 24 is connected to a corresponding adder circuit 31. The output signal appearing on the output terminal of phase shift circuit 24 is given by:

$$V(\theta) = S_{k,j} * V(\theta + \psi_{k,j}) \qquad (1)$$

where

$S_{k,j}$ is the connection strength (gain) of weighting circuit $C_{k,j}$ provided by linear amplifier 23; and

$\psi_{k,j}$ is the phase shift introduced by phase shift circuit 24.

Referring to FIGS. 1 and 2, PLL neural network 20 is a dynamic system that is described mathematically as

$$d\theta_k(t)/dt = \Omega + V(\theta_k)\sum_{j=1}^{n} S_{kj}*V(\theta_j-\pi/2) \qquad (2)$$

for k=1,...,N, where:

θ_k is the phase of the VCO embedded in the k^{th} PLL circuit;

θ_j is the phase of the VCO embedded in the j^{th} PLL circuit;

Ω is the natural frequency of the VCO in MegaHertz (MHz);

S_{kj} are the connection strengths; and

$V(\theta)$ is a 2π periodic waveform function.

PLL neural network computer 20 has an arbitrary waveform function, V, that satisfies "odd-even" conditions and if connection strengths S_{kj} are equal to connection strengths $S_{j,k}$ for all k and j, then the network converges to an oscillatory phase-locked pattern, i.e., the neurons or phase-locked loop circuits oscillate with equal frequencies and constant, but not necessarily zero, phase relationships. Thus, the phase relationships between the oscillators can be used to determine the connection strengths of a neural network computer.

An example of using phase relationships to train neural network computer 20 is described with reference to FIGS. 1, 2, and 3. FIG. 3 illustrates three patterns to be memorized that correspond to the images or symbols "0", "1", and "2," and which are identified by reference numbers 41, 42, and 43, respectively. The phase relationships of a set of key vectors (ξ^m) or patterns of the images are initially memorized using a well known learning rule such as, for example, the Hebbian learning rule. (Other suitable learning rules include the back propagation learning rule, the template learning rule, the least squares learning rule, the correlation learning rule, the perceptron learning rule as well as other supervised and nonsupervised learning rules). In accordance with the Hebbian learning rule, the images are described as a set of key vectors:

$$\xi^m = (\xi^m_1, \xi^m_2, \ldots, \xi^m_N), \quad \xi^m_k = \pm 1, \quad m=0, \ldots, r, \text{ and } k=1,\ldots,N \qquad (3)$$

where

m identifies the pattern to be memorized;
r+1 is the number of patterns to be recognized; and
N is the number of phase-locked loop circuits, i.e., the number of neurons.

Still referring to FIG. 3, patterns "0", "1", and "2" are partitioned into sixty (N=60) pixels or subunits, P_1, P_2, ..., P_{60}, which are described by key vectors ξ^0, ξ^1, ξ^2, respectively, which key vectors are to be memorized or recognized by oscillatory neural network computer 20. For the image or symbol "1," ξ^1_1 is the vector component of the first pixel, P_1; ξ^1_2 is the vector component of the second pixel, P_2; ξ^1_3 is the vector component of the third pixel, P_3; ξ^1_4 is the vector component of the fourth pixel, P_4, etc. It should be noted that in this example, vector component ξ^1_1 describes a white pattern, vector component ξ^1_2 describes a white pattern, vector component ξ^1_3 describes a black pattern, vector component ξ^1_4 describes a black pattern, etc. Key vectors are also determined for the images "0" and "2." When $\xi^m_k = \xi^m_j$, the k^{th} and the j^{th} oscillators are in phase, i.e., $\varphi_k = \varphi_j$, and when $\xi^m_k = -\xi^m_j$, the k^{th} and the j^{th} oscillators are anti-phase, i.e., $\varphi_k = \varphi_j + \pi$. It should be noted that the number of pixels into which the images are partitioned is not a limitation of the present invention.

The key vectors are used in conjunction with the learning rule to determine the connection coefficients of oscillatory neural network computer 20. In the example of using the Hebbian learning rule to memorize the images, the connection coefficients, $S_{k,j}$, are given by:

$$S_{k,j} = (1/n)\sum_{m=0}^{m=r} \xi^m_k \, \xi^m_j \qquad (3)$$

An advantage of using the Hebbian learning rule to determine the connection coefficients is that it produces symmetric connections $S_{k,j}$ so that the network always converges to an oscillatory phase-locked pattern, i.e., the neurons oscillate with equal frequencies and constant, but not necessarily zero, phase relations. It should be understood that some information about each memorized image is included in each connection coefficient.

After the initial strengths are memorized, neural network computer 20 is ready for operation, which operation is described with reference to FIGS. 1-8.

FIG. 4 illustrates a pattern 44 to be recognized which is a degraded or distorted version of the image "1" illustrated in FIG. 3 and previously memorized. In operation, the pattern to

be recognized, i.e., pattern 44, is parsed into a desired number of pixels. Each pixel will correspond to a PLL circuit. Each PLL circuit is the equivalent of a neuron, in our use of the term. In the example shown in FIGS. 4-8 pattern 44 is parsed into sixty pixels, P_1-P_{60}. Thus, oscillatory neural network computer 20 will contain sixty PLL circuits, 25_1-25_{60}. At time t=0, oscillatory neural network computer 20 is initialized such that the initial signals on PLL circuits 25_1-25_{60} represent degraded pattern 44. By way of example, PLL circuits 25_1-25_{60} are initialized by applying an external signal at input terminals IN_1, ..., IN_N of the form:

$$I_k(t) = A_k\cos(\Omega t + \phi_0) \qquad (4)$$

for k=1,...,60, where:

Ω is the same as the center frequency of the PLL;

ϕ_0 is an arbitrary constant;

A_k are large numbers that are positive if the input for the k^{th} channel is to be initialized at +1 and are negative if the k^{th} channel is to be initialized at a -1.

After an initialization interval, the external inputs are turned off and the network proceeds to perform its recognition duties.

Another suitable method for initializing neural network computer 20 is to start the PLL circuits of PLL 25 such that they have different phases that represent pattern 44. Yet another suitable method is to start the PLL circuits of PLL 25 such that they have the same phase and then shift the phase in accordance with pattern 44. Yet another suitable method is to set the initial voltages of the loop filters associated with each PLL circuit of PLL 25. It should be understood that the method for initializing oscillatory neural network computer 20 is not a limitation of the present invention.

Still referring to FIG. 4, at time t=0, pattern 44 is such that the image or value for the pixel P_1 associated with key vector ξ^1_1 is white, the image or value for the pixel P_2 associated with key vector ξ^1_2 is black, the image or value of the pixel P_3 associated with key vector ξ^1_3 is black, the image or value of the pixel P_4 associated with key vector ξ^1_4 is white, the image or value of the pixel P_5 associated with key vector ξ^1_5 is white, the image or value of the pixel P_6 associated with key vector ξ^1_6 is white, the image or value of the pixel P_7 associated with key vector ξ^1_7 is white, the image or value of the pixel P_8 associated with key vector ξ^1_8 is black, ..., the image or value of the pixel P_{60} associated with key vector ξ^1_{60} is black. It should be

understood that the superscripted number 1 indicates this is the phase pattern being recognized, as opposed to a "0" or a "2."

FIG. 5 is a plot 50 of the frequency 51 and the phase 52 portions of output signals, $V(\theta_1)$ and $V(\theta_2)$, of PLL circuits 25_1 and 25_2, respectively, (FIG. 1). At time t=0, the amplitude of output signal $V(\theta_1)$ is +1 and the phase φ_1 is approximately 120 degrees ($2\pi/3$) which corresponds to the white image associated with key vector ξ^1_1; whereas at time t=0 the amplitude of output signal $V(\theta_2)$ is -1 and the phase φ_2 is approximately 0 degrees which corresponds to the black image associated with key vector ξ^1_2.

FIG. 6 is a plot 55 of the frequency 56 and the phase 57 portions of output signals, $V(\theta_1)$ and $V(\theta_3)$, of PLL circuits 25_1 and 25_3, respectively, (FIG. 1). At time t=0, the amplitude of output signal $V(\theta_1)$ is +1 and the phase φ_1 is approximately 120 degrees ($2\pi/3$) which corresponds to the white image associated with key vector ξ^1_1; whereas at time t=0 the amplitude of output signal $V(\theta_3)$ is -1 and the phase φ_3 is approximately 0 degrees which corresponds to the black image associated with key vector ξ^1_3.

Although not shown, it should be understood that there are corresponding output signals $V(\theta_4)$, ..., $V(\theta_{60})$ that occur for each of the respective PLL circuits 25_4, ..., 25_{60}.

Plots 50 and 55 further illustrate pattern recognition in accordance with an embodiment of the present invention. Because the patterns of the individual pixels that have been learned are either black or white, output signals $V(\theta_1)$, ..., $V(\theta_{60})$ lock in phase or in anti-phase to each other depending on the pattern being recognized. For example, in the pattern for a "1" (FIG. 3) pixels P_1, P_2, P_5, and P_6 are white and pixels P_3 and P_4 are black. Thus, when this pattern is recognized from pattern 44, PLL circuits 25_1, 25_2, 25_5, and 25_6 should lock in phase to each other and PLL circuits 25_3 and 25_4 should lock in phase to each other. Further, the PLL circuits for pixels that are of opposite color should lock in anti-phase to each other, i.e., pixels that are white should lock in anti-phase or out of phase to pixels that are black. This is illustrated for PLL circuits 25_1 and 25_2 in FIG. 5 and for PLL circuits 25_1 and 25_3 in FIG. 6. Referring to FIG. 5, output signals $V(\theta_1)$ and $V(\theta_2)$ have a substantially constant in-phase relationship to each other by time t=8.0, i.e., the difference in their phases (φ_1-φ_2) is less than 30 degrees. It should be understood that an acceptable error value for the phase difference of a signal locked in-phase is a design choice. It should be further understood that

the units for time are not included for the sake of clarity. Thus, the units for the time may be seconds, milliseconds, microseconds, etc.

Briefly referring to FIG. 6, output signals $V(\theta_1)$ and $V(\theta_3)$ have a substantially constant anti-phase relationship to each other by time t=8.0, i.e., the difference in their phases $(\varphi_1-\varphi_3)$ is greater than approximately 150 degrees. It should be understood that an acceptable error value for the phase difference of a signal locked in anti-phase is a design choice.

FIG. 7 illustrates the visual outputs for pixels $P_1, \ldots, P_{60}$ from time t=8.0 to time t=9.2. At times t=8.0, 8.6, and 9.2, pixels P_1 and P_2 are white, whereas at times t=8.3, 8.9, and 9.5 pixels P_1 and P_2 are black. It is expected that at times t=8.0, 8.6, and 9.2, the output signals for pixels P_1 and P_2 would have substantially the same amplitude and polarity and be in phase and at times t=8.3, t=8.9, and t=9.5, the output signals for pixels P_1 and P_2 would have substantially the same amplitude and polarity and be in phase. Comparing these times with the output signals shown in FIG. 5, at times t=8.0, t=8.6, and t=9.2 output signals $V(\theta_1)$ and $V(\theta_2)$ have an amplitude of +1 and are substantially in phase. Likewise, at times t=8.3, t=8.9, and t=9.5 output signals $V(\theta_1)$ and $V(\theta_2)$ have an amplitude of -1 and are substantially in phase.

Further, it is expected that at times t=8.0, 8.3, 8.6, 8.9, 9.2, and 9.5, the output signals for pixels P_1 and P_3 would be in anti-phase and have amplitudes of substantially the same magnitude but opposite polarity. Comparing these times with the output signals shown in FIG. 6, at times t=8.0, 8.6, and 9.2 output signal $V(\theta_1)$ has an amplitude of +1 and $V(\theta_3)$ has an amplitude of -1, and at times t=8.3, t=8.9, and t=9.5 output signal $V(\theta_1)$ has an amplitude of -1 and $V(\theta_3)$ has an amplitude of +1.

The output signals of neural network computer 20 can be monitored by multiplying each output signal with a reference output signal. In the example of recognizing pattern 44 from FIG. 4, the output signals of each of pixels $P_1, \ldots, P_{60}$ are multiplied with that of pixel P_1. The result of this multiplication is a new sixty pixel image where the image of new pixel P_{1new} is the product of output $V(\theta_1)$ with itself; the image of new pixel P_{2new} is the product of output signals $V(\theta_1)$ and $V(\theta_2)$; the image of new pixel P_{3new} is the product of output signals $V(\theta_1)$ and $V(\theta_3)$; the image of new pixel P_{4new} is the product of output signals $V(\theta_1)$ and $V(\theta_4)$, etc.

This result is illustrated in FIG. 8 where the products are shown at time intervals t=0, t=1, t=2, …, t=10.

By now it should be appreciated that a method for recognizing patterns and an oscillatory neural network computer for implementing the method have been provided. An important aspect of this invention is the discovery by the inventors that the output signals for a PLL neural network computer oscillate with equal frequencies and constant, but not necessarily zero, phase relationships. Thus, the phase relationships of the neural network computer are used to determine the connection strengths of the neural network computer. This provides an increased immunity to noise. Another advantage of the present invention is that the type of learning rule used to train the neural network computer is not a limitation of the present invention.

Although certain preferred embodiments and methods have been disclosed herein, it will be apparent from the foregoing disclosure to those skilled in the art that variations and modifications of such embodiments and methods may be made without departing from the spirit and scope of the invention. Accordingly, it is intended that the invention shall be limited only to the extent required by the appended claims and the rules and principles of applicable law.